Addison-Wesley.

Wolpert, D. (1994b). On the Bayesian “Occam factors” argument for Occam’s razor. To appear in *Computational Learning Theory and Natural Learning Systems: Volume III Natural Learning Systems*, S. Hanson et al. (Ed.’s). MIT Press.

Wolpert, D. and Macready, W. (1994). No Free Lunch Theorems for Search. SFI TR 95-02-010. Submitted.

Kearns, M. J., et al. ‘Towards efficient agnostic learning, in *Proceedings of the 5th annual workshop on Computational Learning Theory*, ACM Press, NY, NY, 1992.

Knill, M, Grossman, T., and Wolpert, D. (1994). Off-Training-Set Error for the Gibbs and the Bayes Optimal Generalizers. Submitted.

Mitchell T., Blum, A. (1994). *Course notes for Machine Learning*, CMU.

Murphy, P., Pazzani, M. (1994). Exploring the decision forest: an empirical investigation of Occam’s razor in decision tree induction. *Journal of Artificial Intelligence Research*, **1**, 257-275.

Natarajan, B. (1991). *Machine Learning: A theoretical approach*. Morgan Kauffman, San Mateo, CA.

Perrone, M. (1993). Improving regression estimation: averaging methods for variance reduction with extensions to general convex measure optimization. Ph.D. thesis, Brown Univ. Physics Dept.

Plutowski, M., et al. (1994). Cross-validation estimates integrated mean squared error. In *Advances in neural information processing systems 6*, Cowan et al. (Ed.’s), Morgan Kauffman, CA.

Schaffer, C., (1993). Overfitting avoidance as bias. *Machine Learning*, **10**, 153-178.

Schaffer, C. (1994). A conservation law for generalization performance. In Cohen and Hirsh (Ed.’s), *Machine Learning: Proceedings of the Eleventh International Conference*. Morgan Kauffman, San Francisco.

Schapire, R. (1990). The strength of weak learnability. *Machine Learning*. **5**, 197-227.

Vapnik, V. (1982). Estimation of dependences based on empirical data. Springer-Verlag.

Vapnik, V., and Bottou, L. (1993). Local algorithms for pattern recognition and dependencies estimation. *Neural Computation*, **5**, 893-909.

Waller, W., Jain, A. (1978). On the monotonicity of the performance of Bayesian classifiers. *IEEE Transactions on Information Theory*, **IT-24**, 392-394.

Weiss, S.M., and Kulikowski, C. A. (1991). *Computer systems that learn*. Morgan Kauffman.

Wolpert, D. (1992). On the connection between in-sample testing and generalization error. *Complex Systems*, **6**, 47-94.

Wolpert, D. (1993). On overfitting avoidance as bias. SFI TR 93-03-016.

Wolpert, D. (1994a). The relationship between PAC, the Statistical Physics framework, the Bayesian framework, and the VC framework. In *The Mathematics of Generalization*. D. Wolpert (Ed.).

$$P(c \mid y_H, q, d) = \Sigma_{y_F} \delta(c, L(y_H, y_F)) \int df(x \notin d_X)\ f(q, y_F) \int df(x \in d_X)\ P(d \mid f)\ / P(d).$$

Up to overall proportionality constants, we can now replace both integrals with $\int df$. By the same reasoning using in the proof of theorem (1), the first integral is a constant. If we reintroduce the constant P(f) into the remaining integral, we get $\int df\ P(d \mid f)\ P(f) / P(d) = 1$. Therefore, $P(c \mid y_H, q, d) = \Sigma_{y_F} \delta(c, L(y_H, y_F))$. (Note that for homogenous L(., .), this is independent of y_H.) As needed, this is independent of q and d. QED.

References

Anthony M. and Biggs N. (1992). *Computational Learning Theory*. Cambridge University Press.

Berger, J. (1985). *Statistical decision theory and Bayesian analysis.* Springer-Verlag.

Bernardo, J. Smith, A. (1994). *Bayesian Theory*. Wiley and Sons, NY.

Berger, J., and Jeffreys, W. (1992). Ockham's razor and Bayesian analysis. *American Scientist*, **80**, 64-72.

Blumer, A., et alia (1987). Occam's razor. *Information Processing Letters*, **24**, 377-380.

Blumer, A., et alia (1989). Learnability and Vapnik-Chervonenkis dimension. *Journal of the ACM*, **36**, 929-965.

Bridle, J. (1989). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fougelman-Soulie and J. Herault (Eds.), *Neuro-computing: Algorithms, architectures, and applications.* Springer-Verlag.

Dietterich, T. (1990). Machine Learning. Annu. Rev. Comput. Sci., **4**, 255-306.

Drucker, H. et al. (1993). Improving performance in neural networks using a boosting algorithm. In *Neural Information Processing Systems 5,* S. Hanson et al. (Eds). Morgan-Kauffman.

Duda, R., and Hart, P. (1973). *Pattern classification and scene analysis.* Wiley and Sons.

Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, **IT-14**, 55-63.

similarly for $s_{max}(\Xi, \phi)$. Then what we need is for there to be an h, h^*, such that

$$\{\varepsilon - s_{max}(d_X, \phi)\} / \pi(X - d_X) \; < \; E(C \mid \phi, h^*, d) \; \leq \; \{\varepsilon - s_{min}(d_X, \phi)\} / \pi(X - d_X).$$

In particular, for zero-one loss, and single-valued f, $s_{min}(d_X, \phi) = 0$, and $s_{max}(d_X, \phi) = \pi(d_X)$. So we need an h^* such $\{\varepsilon - \pi(d_X)\} / \pi(X - d_X) \; < \; E(C \mid \phi, h^*, d) \; \leq \; \varepsilon / \; \pi(X - d_X)$. For any $\varepsilon < 1$, there is such an h^*. This establishes theorem (14).

APPENDIX C. Proof of theorem (15)

We need to prove that for a vertical likelihood and OTS error, for uniform P(f), all learning algorithms with the same $P(y_H \mid m)$ have the same P(c | m). First write $P(c \mid m) = \Sigma_{y_H} P(c \mid y_H, m) \quad P(y_H \mid m)$ (where from now on the uniformity of P(f) is implicit). If $P(c \mid y_H, m)$ is independent of the learning algorithm, the proof will be complete. So expand $P(c \mid y_H, m) = \Sigma_{q,d} P(q, d \mid y_H) P(c \mid y_H, q, d)$. I will show that $P(c \mid y_H, q, d)$ is a fixed function of c and y_H that is independent of q and d. This in turn means that $P(c \mid y_H, m)$ is the same fixed function of c and y_H, which is exactly the result we need.

Write $P(c \mid y_H, q, d) = \int df \, \Sigma_{y_F} \delta(c, L(y_H, y_F)) f(q, y_F) P(f \mid y_H, q, d)$. Next use Bayes' rule to rewrite $P(f \mid y_H, q, d)$ as $P(y_H \mid f, q, d) P(f \mid q, d) / P(y_H \mid q, d)$.

Now write $P(y_H \mid q, d) = \Sigma_H h(q, y_H) P(h \mid d)$. But $P(y_H \mid f, q, d)$ is given by the same sum. Therefore $P(f \mid y_H, q, d) = P(f \mid q, d)$. Using Bayes' theorem again, $P(f \mid q, d) = P(q \mid d, f) P(d, f) \; / \; [P(q \mid d) P(d)] = P(d \mid f) P(f) / P(d) \propto P(d \mid f) / P(d)$ (since we have a uniform P(f)). So combining everything, up to an overall proportionality constant,

$$P(c \mid y_H, q, d) = \Sigma_{y_F} \delta(c, L(y_H, y_F)) \int df \; f(q, y_F) P(d \mid f) / P(d).$$

Now recall that $P(c \mid y_H, q, d)$ is occurring after being multiplied $P(q, d \mid y_H)$. For OTS error, $P(q, d \mid y_H) = 0$ unless $q \in X - d_X$. Therefore in evaluating $P(c \mid y_H, q, d)$ we can assume that $q \in \; X - d_X$. Accordingly, the term $f(q, y_F)$ only depends on the values of $f(x \notin d_X)$. Since we have a vertical likelihood, the P(d | f) term only depends on $f(x \in d_X)$. Accordingly, we can write

value of $S(h_1, \phi, d_X)$ as s_1, and a value of $S(h_2, \phi, d_X)$ as s_2. By (C.1), the set of h_1 under consideration are those for which $\varepsilon \geq s_1$, and such that for OTS error C,

C.2) $E(C \mid h_1, \phi, d) \leq \{\varepsilon - s_1\} / \pi(X - d_X)$,

and similarly for $E(C \mid h_2, \phi, d)$. (The expression "$\pi(\Xi)$" is shorthand for $\Sigma_{x \in \Xi} \pi(x)$.)

We are interested in $\Sigma_{H_1, H_2 \in H_\phi(\varepsilon)} E(C \mid h_1, h_2, \phi, d, i)$, where 'i' is either A or B. As usual, expand the sum into an outer and inner sum, getting

$\Sigma_{H_1(x \in d_X), H_2(x \in d_X)} \Sigma_{H_1(x \notin d_X), H_2(x \notin d_X)} E(C \mid h_1, h_2, \phi, d, i)$,

where the restriction that both h_1 and h_2 lie in $H_\phi(\varepsilon)$ is implicit.

Consider the inner sum, for any particular set of values of $h_1(x \in d_X)$ and $h_2(x \in d_X)$. Without loss of generality, say that for the d at hand and the values of $h_1(x \in d_X)$ and $h_2(x \in d_X)$, strategy A picks h_1. This means that $s_1 \leq s_2$.

Examine the case where s_1 is strictly less than s_2. Using (C.2), the inner sum over $h_1(x \notin d_X)$ is over all $h(x \notin d_X)$ such that $E(C \mid \phi, h, d) \leq \{\varepsilon - s_1\} / \pi(X - d_X)$, and the inner sum over $h_2(x \notin d_X)$ is over all $h_2(x \notin d_X)$ such that $E(C \mid \phi, h, d) \leq \{\varepsilon - s_2\} / \pi(X - d_X)$.

Since $s_1 < s_2$, this means that the h's going into the sum over $h_1(x \notin d_X)$ are a proper superset of the h's going into the sum over $h_2(x \notin d_X)$. In addition, for all the h's that are in the h_1 sum but not in the h_2, $E(C \mid \phi, h, d) \geq 0$. (In fact, those h's obey $\{\varepsilon - s_2\} / \pi(X - d_X) < E(C \mid \phi, h, d) \leq \{\varepsilon - s_1\} / \pi(X - d_X)$.) Therefore so long as the set $H^*(s_1, s_2)$ of h's in the h_1 sum but not in the h_2 sum is not empty, the h_1 sum is larger than the h_2 sum.

This means that for all cases where s_1 is strictly less than s_2 and $H^*(s_1, s_2)$ is non-empty, $\Sigma_{H_1(x \notin d_X), H_2(x \notin \in d_X)} E(C \mid h_1, h_2, \phi, d, i)$ is larger for algorithm A than for algorithm B. If $s_1 = s_2$, then $\Sigma_{H_1(x \notin d_X), H_2(x \notin \in d_X)} E(C \mid h_1, h_2, \phi, d, i)$ is the same for both algorithms. So if there are any h_1 and h_2 in the sum such that the associated $H^*(s_1, s_2)$ is non-empty, it follows that $\Sigma_{H_1, H_2 \in H_F(\varepsilon)} E(C \mid h_1, h_2, \phi, d, i)$ is larger for algorithm A than for algorithm B. If there are no such h_1 and h_2, $\Sigma_{H_1, H_2 \in H_F(\varepsilon)} E(C \mid h_1, h_2, \phi, d, i)$ is the same for both algorithms. This establishes theorem (13).

To understand when there are any h_1 and h_2 such that the associated $H^*(s_1, s_2)$ is non-empty (so that we get a strict inequality in theorem (13)), make the definition $s_{min}(\Xi, \phi) \equiv \Sigma_{q \in \Xi} \pi(q) \min_{y_H}(L(y_H, \phi(q)))$, and

theorem (1) of paper one. More formally, follow the reasoning presented after lemma (1), only applying it to our distribution rather than to $\int df\ P(c \mid f, d)$. The result is lemma (3) (rather than theorem (1)). QED.

Proof of theorem (11): Expand our sum as

$$\Sigma_{h_1(x \in d_X), H_2(x \in d_X)}\ \Sigma_{h_1(x \notin d_X), H_2(x \notin d_X)} P(c \mid f, h_1, h_2, d, i).$$

Consider any set of values of $h_1(x \in d_X)$, $h_2(x \in d_X)$ such that the strategy at hand picks h_1. For such a set of values, our inner sum becomes (up to an overall proportionality constant determined by m') $\Sigma_{h_1(x \notin d_X)} P(c \mid f, h_1, d)$, with obvious notation. By lemma (3), this is simply some function $\lambda(c)$, independent of f and d. The same is true for those sets of values of $h_1(x \in d_X), h_2(x \in d_X)$ such that the strategy at hand picks h_2. Therefore, up to overall constants, $\Sigma_{h_1, h_2} P(c_i \mid f, h_1, h_2, d)$ is just $\lambda(c)$. This is true regardless of which strategy we use. We have now established the following.

Proof of theorem (16): For a uniform P(f), $P(y_H \mid m) \propto \Sigma_{q,d} \int df\ dh\ h(q, y_H)\ P(h \mid d)\ P(q \mid d)\ P(d \mid f)$. The integral over f only works on the P(d | f). For the likelihood at hand, it is some function func(m, m'), independent of d. Therefore we have

$$P(y_H \mid m) \propto \Sigma_{q,d} \int dh\ \ func(m, m')\ h(q, y_H)\ P(h \mid d)\ P(q \mid d_X)$$

$$= \Sigma_{q, d_X} \int dh\ \ func(m, m')\ h(q, y_H)\ P(q \mid d_X)\ \Sigma_{d_Y}\ \ P(h \mid d_X, d_Y).$$

By hypothesis, the sum over d_Y is the same for our two algorithms. QED.

APPENDIX B. PROOF OF THEOREMS (13) AND (14)

Let $H_\phi(\varepsilon)$ indicate the set of h's such that $E(C_{IID} \mid h, \phi) \leq \varepsilon$. This appendix analyzes the sign of $\Sigma_{H_1, H_2 \in H_\phi(\varepsilon)}$ [E(C | h_1, h_2, ϕ, d, A) - E(C | h_1, h_2, ϕ, d, B)] for OTS error. In particular, it shows that this quantity is necessarily positive for zero-one loss for any $\varepsilon < 1$.

We can express $H_\phi(\varepsilon)$ as the set of h's such that

C.1) $\Sigma_{q \in d_X} L(h(q), \phi(q))\ \pi(q)\ + \Sigma_{q \notin d_X} L(h(q), \phi(q))\ \pi(q) \quad \leq \quad \varepsilon.$

It is useful to introduce the following notation. Define $S(h, \phi, \Xi) \equiv \Sigma_{q \in \Xi} L(h(q), \phi(q))\ \pi(q)$. Indicate a

break the integral over f into two integrals, one over the values of $f(x \in d_X)$, and one over the values of $f(x \notin d_X)$. The integral over $f(x \notin d_X)$ transforms the $f(q, y_F)$ term into a constant, independent of q and y_F. We can then bring back in the P(f), and replace the integral over $f(x \in d_X)$ with an integral over all f (up to overall proportionality constants). So up to proportionality constants, we're left with

$P(y_H \mid y_F, m) = \Sigma_{d,q} \int dh\, h(q, y_H)\, P(h \mid d)\, P(q \mid d)\, P(d) / P(y_F \mid m)$.

Now write $P(y_F \mid m) = \Sigma_{d,q} \int df\, P(y_F \mid f, d, q)\, P(f, d, q \mid m) \propto \Sigma_{d,q} \int df\, f(q, y_F)\, P(q \mid d)\, P(d \mid f)$. As before, the P(q | d) allows us to break the integral over all f into two integrals, giving us some overall constant. So $P(y_H \mid y_F, m) \propto \Sigma_{d,q} \int dh\, h(q, y_H)\, P(h \mid d)\, P(q \mid d)\, P(d)$. However this is exactly the expression we get if we write out $P(y_H \mid m)$. QED.

6. As an aside, consider the case where $Y = \{1, 2, ..., r\}$ and r is odd. Now $E((y_H)^2 \mid m) - 2\, E(y_H \mid m)\, E(y_F \mid m) = -[E(y_F \mid m)]^2 + E((y_H - E(y_F \mid m))^2 \mid m)$. So guessing such that y_H always equals $E(y_F \mid m)$ gives best behavior. In particular, for the likelihood of equation (1.1), $E(y_F \mid m) = \Sigma^r_{i=1} i / r = (r + 1) / 2$, so best behavior comes from guessing halfway between the lower and upper limits of Y.

Similarly, $E((y_H)^2 \mid m) = Var(y_H \mid m) + (E(y_H \mid m))^2$ (with the implicit notation that "Var(a | b)" is the variance of a, for probabilities conditioned on b). So for two learning algorithms A and B with the same $E(y_H \mid m)$, guessing the algorithm with the smaller variance is preferable. These results justify the claims made at the beginning of this section, for the case of a uniform prior.

APPENDIX A. MISCELANEOUS PROOFS

Proof of lemma (3): Expand the sum in question as

$\Sigma_{h(x \notin d_X),q,y_H,y_F} \delta(c, L(y_H, y_F))\, P(q \mid d)\, f(q, y_F)\, h(q)$.

Since $P(q \mid d) = 0$ for $q \in d_X$, this sum is independent of the values of $h(x \in d_X)$. Accordingly, up to an overall constant, we can rewrite it as

$\Sigma_{h,q,y_H,y_F} \delta(c, L(y_H, y_F))\, P(q \mid d)\, f(q, y_F)\, h(q)$.

We can use the reasoning of the NFL theorems to calculate this. Intuitively, all we need do is note by lemma (1) (see paper one) that our sum equals $\int df\, P(c \mid f, d)$ if one interchanges f and h, and then invoke

Indeed, by the argument just below lemma (1) in paper one, we know that for the random learning algorithm $E(C \mid f, m) = \Sigma_c \Lambda(c)\, c / r$ for all f, so that $\max_f E(C \mid f, m) = \min_f E(C \mid f, m)$ for that algorithm. Using the NFL theorems, this means that the random learning algorithm is minimax superior to all other learning algorithms.

2. It's important to keep mind though that even if head-to-head minimax behavior does end up playing *a priori* favorites between algorithms, it's not clear why one should care about such behavior rather than simple expected cost (which by the NFL theorems plays no such favorites). One intriguing possible answer to this question is that in choosing between species A and species B (and their associated organic learning algorithms), natural selection may use head-to-head minimax behavior. The idea is that for those targets f for which A's behavior is just slightly preferred over B's, equilibrium has a slightly smaller population for species A than for species B. But if there is any non-linearity in the system, then for any f's for which A's behavior is far worse than B's, A goes extinct. Therefore if over time the environment presents A and B with a series of f's, the surviving species will be the one with preferable head-to-head minimax behavior.

3. In terms of appendix B, we still get $s_1 < s_2$. In appendix B this meant that the upper limit on h_2's error over $X - d_X$ was less than the upper limit on h_1's error, while they shared lower limits. Here it instead means that the lower limit on h_2's error over $X - d_X$ is lower than the lower limit on h_1's error, while they share the same upper limit.

4. Phrased differently, for a quadratic loss function, given a series of experiments and a set of deterministic learning algorithms G_i, it is always preferable to use the average generalizer $G' \equiv \Sigma_i G_i / \Sigma_i 1$ for all such experiments rather than to randomly choose a new member of the G_i to use for each experiment. Intuitively, this is because such an average reduces variance without changing bias. (See [Perrrone 1993] for a discussion of what is essentially the same phenomenon, in a neural net context.) Or to put it another way, no matter what value a real-number "truth" z has, if one has two real numbers α and β, then it is always true that $(z - [\alpha + \beta] / 2)^2 \leq (z - \alpha)^2 / 2 + (z - \beta)^2 / 2$. Note though that this effect in no way implies that using G' for all the experiments is better than using any single particular $G \in \{G_i\}$ for all the experiments.

5. To see this, write $P(y_H \mid y_F, m) = \Sigma_{d,q} \int dfdh\, h(q, y_H)\, P(h \mid d)\, P(d, q, f \mid y_F, m)$. Use Bayes' theorem to write $P(d, q, f \mid y_F, m) = P(y_F \mid f, d, q)\, P(f, d, q \mid m) / P(y_F \mid m) = f(q, y_F)\, P(q \mid d)\, P(d \mid f)\, P(f) \;/\; P(y_F \mid m)$. Combining, $P(y_H \mid y_F, m) = \Sigma_{d,q} \int dfdh\, h(q, y_H)\, P(h \mid d)\, f(q, y_F)\, P(q \mid d)\, P(d \mid f)\, P(f) \;/\; P(y_F \mid m)$.

In the usual way, we take P(f) to be a constant, have the P(q | d) restrict q to lie outside of d_X, and then

tioned as “beyond the scope of this paper” in section 4 of paper one.

12) Investigate what priors P(f) have $E(_{COTS} \mid \text{no punt}, m) \leq E(C_{OTS} \mid \text{punt}, m)$.

13) Investigate if and how results change if one conditions on a value of m' rather than m.

14) Carry through the analysis for error C' rather than C.

15) Investigate what set of conditions give NFL results for any algorithm in an equivalence class of algorithms that are scrambled versions of one another (as opposed to NFL results for all algorithms). As an example, it seems plausible that any $P(\phi)$ for which each $\phi(x)$ is determined by IID sampling some distribution $R(y)$ - i.e., any $P(\phi) = \Pi_x R(\phi(x))$ - results in NFL-style equivalence between all algorithms in any scramble-based equivalence class of algorithms. (The restricting to the equivalence class is necessary since an algorithm with a disposition to guess argmax(R(y)) will likely outperform one that tries to guess argmin(R(y)).) Similarly, it seems plausible that if one keeps f fixed but averages over all bijections of X to itself - i.e., if one averages over all encodings of the inputs - then all algorithms in such an equivalence class have the same generalization performance. (This latter result would constitute a sort of NFL-for-feature-representations result.)

Acknowledgments: I would like to thank Cullen Schaffer, Wray Buntine, Manny Knill, Tal Grossman, Bob Holte, Tom Dietterich, Mark Plutowski, Karl Pfleger, Joerg Lemm, Bill Macready and Jeff Jackson for interesting discussions. This work was supported in part by the Santa Fe Institute and by TXN Inc.

FOOTNOTES

1. To simply say that A is minimax superior to B without the “head-to-head” modifier would imply instead something like $\max_f E(C \mid f, m, A) \leq \max_f E(C \mid f, m, B)$. Such minimax superiority is of little interest.

2) Investigate distributions of the form $P(c_1, c_2 \mid ...)$, where c_i is the error for algorithm i, when there are more than two possible values of the loss function.

3) Investigate the validity of the potential "head-to-head minimax" justification for cross-validation, both for the scenario discussed in this paper, and for non-homogenous loss functions and/or empirical-loss-conditioned distributions, and/or averaging over hypotheses rather than targets.

4) Investigate whether there are scenarios in which one generalizer is head-to-head minimax-superior to the majority of other generalizers. Investigate whether there are "head-to-head minimax superior cycles", in which A is superior to B is ... superior to A, and if so characterize those cycles.

5) Investigate for what distributions over hypotheses h do sums over h's with the target f fixed result in the majority algorithm beating the anti-majority algorithm.

6) Investigate sums over h's with f fixed when f and/or h is not single-valued (note for example that if $f(q, y) = 1 / r \; \forall$ q and y, then all h's have the same value of $\Sigma_{y_H, y_F} f(q, y_F) h(q, y_H) L(y_H, y_F)$ if L(., .) is homogenous);

7) Investigate sums over h's where f too is allowed to vary.

8) Investigate under what circumstances $E(C_{IID} \mid m) < E(C_{OTS} \mid m)$.

9) Find the set of P(f)'s such that all learning algorithms are equivalent, as determined by P(c | m). There are other P(f)'s besides the uniform one for which this is so. An example given in the text is that for fixed homogenous noise, the distribution that is uniform over ϕ's results in NFL. Find the distributions $G(\alpha)$ over priors α such that all algorithms have the same average according to $G(\alpha)$ of $P(c \mid \alpha, m)$. For any pair of algorithms, characterize the distributions $G(\alpha)$ for which the algorithms have the same average according to $G(\alpha)$ of $P(c \mid \alpha, m)$.

10) Find the set of P(f)'s for which two particular algorithms are equivalent according to P(c | m). Find the distributions L(f) such that two particular algorithms have the same average (according to L(f)) of P(c | f m).

11) Investigate the empirical-loss-conditioned distributions and punt-signal-conditioned distributions men-

On the other hand, there may be some assurances associated with non-averaging criteria, like head-to-head minimax criteria. In addition, for certain other noise processes and/or certain loss functions one can, *a priori*, say that one algorithm is superior to another, even in terms of averages.

More generally, for all its reasonableness when stated in the abstract, the full implications of the no-assurances concept for applied supervised learning can be surprising. For example, it implies that there are "as many" targets (or priors over targets) in which any algorithm performs *worse than random* as there are for which it performs better than random (whether one conditions on the training set, on the training set size, on the target, or what have you). In particular it implies that cross-validation fails as readily as it succeeds, boosting makes things worse as readily as better, active learning and/or algorithms that can choose not to make a guess fail as readily as they succeed, etc. All such implications should be kept in mind when encountering quotes like those at the beginning of the introduction of paper one, which taken at face value imply that, even without making any assumptions about the target, one can have assurances that one's learning algorithm generalizes well.

In addition to these kinds of issues, in which the generalizer is fixed and one is varying the target, this second paper also discusses scenarios where the generalizer can vary but the target is fixed. For example, this second paper uses such analysis to show that if one averages over generalizers cross-validation is no better than "anti" cross-validation, *regardless of the target*. In this sense, cross-validation cannot be justified as a Bayesian procedure - no prior over targets justifies its use for all generalizers.

There are many avenues for future research on the topic of OTS error. Restrictions like homogenous noise, vertical likelihoods, fixed (as opposed to random variable) sampling distributions etc., are all imposed to "mod out" certain "biasing" effects in the mathematics. Future work investigates the ramifications of relaxing these conditions, to precisely quantify the associated effects.

As an example, local noise in the observed **X** values will have the effect of "smoothing" or "smearing" the target. With such noise, the likelihood is no longer vertical, and information from the training examples does, *a priori*, leak into those off-training set x that lie near d_X. So one does not get the NFL theorems in general when there is such noise. However one can imagine restricting attention to those learning algorithms that "respect" the degree of smoothness that noise in **X** imposes. Presumably NFL-like results hold among those algorithms, but none of this has been worked out in any detail.

Some other avenues of future research involving OTS error are mentioned in [Knill et al. 1994]. However even if one restricts attention to the limited NFL aspect of OTS error, and even for the restrictions imposed in this pair of papers (e.g., vertical likelihood), there are still many issues to be explored. Some of them are:

1) Characterize when cross-validation beats anti-cross-validation for non-homogenous loss functions.

n / 2}, for cross-validation, there is a 2/3 chance of choosing h_1 (cross-validation will choose h_1 if $d_Y = 2$ or 3, for such a d_X). Similarly, for anti-cross-validation, there is a 2/3 chance of choosing h_2. Now since q must lie outside of d_X, for such a d_X, $E((y_H - 2)^2 \mid m)$ is smaller if y_H is given by h_2 than if it is given by h_1.

So anti-cross-validation does better if $d_X \in \{1, 2, ..., n / 2\}$. For analogous reasons, anti-cross-validation also does better if $d_X \in \{1 + n/2, ..., n\}$. So anti-cross-validation wins regardless of d_X. QED.

Example 8: As an example of where cross-validation beats anti-cross-validation, consider the same scenario as in the preceding example, only with different h_1 and h_2. Have $h_1 = 2$ for all x, and $h_2 = 1$ for all x. Cross-validation is more likely to guess h_1, and anti-cross-validation is more likely to guess h_2, regardless of d_X. However h_1 has a better $E((y_H - E(y_F \mid m))^2 \mid m)$ than does h_2, again, for all d_X. QED.

Note the important feature of the preceding two examples that whether cross-validation works (in comparison to anti-cross-validation) depends crucially on what learning algorithms you're using it to choose among. This is another illustration of the fact that assuming that cross-validation works is not simply making an assumption concerning the target, but rather making an assumption about the relationship between the target and the learning algorithms at hand.

Intuitively, if one uniformly averages over all h_1 and h_2, there is no statistical correlation between the training set and the cost, regardless of whether one uses algorithm A or algorithm B. When the loss function is homogenous, this lack of correlation results in the NFL theorems. When the loss function isn't homogenous, it instead gives the results of this section.

5. CONCLUSIONS AND FUTURE WORK

These two papers investigate some of the behavior of OTS (off-training set) error. In particular, they formalize and investigate the concept that "if you make no assumptions concerning the target, then you have no assurances about how well you generalize".

As stated in this general manner, this no-assurances concept is rather intuitive and many researchers would agree with it readily. However the details of what "no assurances" means are not so obvious. For example, for the conventional loss function, noise process etc. studied in computational learning theory, there are indeed no generalization assurances associated with averages. (As a result, if we know that (for example) averaged over some set of targets, F, the generalizer CART is superior to some canonical neural net scheme, then we also know that averaged over targets not contained in F, the neural net scheme is superior to CART.)

iii) Implications of the equivalence of any learning algorithm with its scrambled version

The results of the preceding subsection tell us that there is no *a priori* reason to believe that there is any value in a particular learning algorithm's relationship between training sets d and resulting hypotheses h. All that matters about the algorithm is how prone it is to guess certain y_H's, not the conditions determining when it makes those guesses.

However these results are not as strong as the NFL theorems. As an example, these results do tell us that cross-validation (used to choose among a pre-fixed set of learning algorithms) and its scrambled version always gives the same uniform f-average of P(c | f, m) (if one has OTS error, etc.). So in that sense there is no *a priori* justification for the d-dependence of cross-validation, even for non-homogenous loss functions. However for non-homogenous loss functions we cannot automatically equate cross-validation with anti-cross-validation, like we could for homogenous loss functions. This is because the two techniques can have different $P(y_H \mid m)$. In fact, there are some scenarios in which cross-validation has a better uniform f-average of P(c | f, m), and some in which anti-cross-validation wins instead. Not only can't one equate the techniques, one can't even say that the relative superiority of the techniques is always the same.

Example 7: As an example of where anti-cross-validation has better uniform f-average of its expected cost than does cross-validation, let both techniques be used to choose between the pair of learning algorithms A and B. Let A be the algorithm "always guess h_1 regardless of the data" and B be the algorithm "always guess h_2 regardless of the data". Let the training set consist of a single element, and let the "validation set" part of the training set be that single element. So cross-validation will guess whichever of h_1 and h_2 is a better fit to the training set, and anti-cross-validation will guess whichever is a worse fit to the training set.

Let Y = {1, 2, 3}, and X = {1, 2, ..., n} where n is even. Let $h_1(x) = 2$ for $x \in \{1, 2, ..., n / 2\}$, and let it equal 1 for the remaining x. Conversely, $h_2(x) = 1$ for $x \in \{1, 2, ..., n / 2\}$, and 2 for the remaining x. Assume we are using the quadratic loss function to measure C, and that both cross-validation and anti-cross-validation use the quadratic loss function to choose between h_1 and h_2. Assume the likelihood of equation (1.1) and a uniform sampling distribution. Use a uniform P(f).

As shown in example (6), for this scenario, $E(C \mid m) = E((y_H - E(y_F \mid m))^2 \mid m)$. For our scenario, for a uniform P(f), $E(y_F \mid m) = 2$. So in comparing the uniform f-average of expected cost for cross-validation with that for anti-cross-validation, it suffices to compare the values of $E((y_H - 2)^2 \mid m)$ for the two techniques.

Now by symmetry, whatever d_X is, d_Y is equally likely to be 1, 2, or 3. Therefore if $d_X \in \{1, 2, ...,$

$\Sigma_f P(c \mid f, m)$, where the proportionality constant is independent of the learning algorithm. This establishes the following corollary of theorem (15).

Corollary 5 Consider two learning algorithms that would have the same $P(y_H \mid m)$ if P(f) were uniform. For a vertical likelihood and OTS error, the uniform average over f of $P(c \mid f, m)$ is the same for those two algorithms.

Now make the following definition.

Definition 1 The learning algorithm B is a "scrambled" version of the learning algorithm A if and only if for all d_X, $\Sigma_{d_Y} P(h \mid d_X, d_Y, B) = \Sigma_{d_Y} P(h \mid d_X, d_Y, A)$. (The sums are implicitly restricted to d_Y's with the same number of elements as d_X.)

Intuitively, if B is a scrambled version of A, then B is identical to A except that A's correlation between its guess h and the (output components of the) data has been scrambled.

As an example, view (a deterministic) algorithm A as a big table of h values, indexed by training sets d. Then a scrambled version of A is given by the same table where the entries within each block corresponding to a particular d_X have been permuted.

Note that if for certain training sets A creates h's with low training set error, then in general a scrambled version of A will have much higher error on those training sets. Note also that scrambling does **not** involve randomizing the h's the algorithm can guess. It doesn't touch the set of h's. Rather scrambling involves randomizing the rule for which h is associated with which d.

The following result in proven in appendix A:

Theorem 16 Assume the (vertical) likelihood of equation (1.1). Then if learning algorithm B is a scrambled version of A, it follows that for uniform P(f), $P(y_H \mid B, m) = P(y_H \mid A. m)$.

Combining theorem (16) and corollary (5), we see that for the likelihood of equation (1.1) and OTS error, if algorithm B is a scrambled version of algorithm A, then the algorithms have the same uniform average over f of $P(c \mid f, m)$. This constitutes an NFL theorem relating an algorithm to its scrambled version.

Note that if the sampling distribution $\pi(x)$ is uniform, we can relax the definition of scrambling to simply have $\Sigma_d P(h \mid d, A) = \Sigma_d P(h \mid d, B)$ and theorem (16) will still hold. (For such a case $P(q \mid d_X)$ will be a function purely of m' that can be absorbed into the quantity func(m, m') discussed in appendix A.)

This "scrambling" tool can also be used to help analyze the case where non-homogenous noise-processes are superimposed on single-valued target functions (recall that some of the NFL theorems require homogeneity of such noise). For reasons of space though, such an analysis is not presented here.

ii) The equivalence of any learning algorithm with its scrambled version

For reasons of space, I will only consider the case where all targets f are allowed (rather than the case where are f's expressible as some ϕ with noise superimposed are allowed). Since the focus is on how (if at all) the statistical relationship between d and h embodied in the generalizer correlates with error, we must allow the training set d to vary. Accordingly, the results are conditioned on training set size m rather than on the precise training set d at hand.

The following theorem, which holds regardless of the loss function, is proven in appendix C.

Theorem 15 For a vertical likelihood, uniform P(f), and OTS error, all learning algorithms with the same $P(y_H \mid m)$ have the same $P(c \mid m)$.

Example 6: This example presents an explicit corroboration of theorem (15) in a special case. Say we have quadratic loss L(., .). In general, we will only be interested in the case where r > 2. (For binary Y, e.g., Y = {0, 1}, quadratic loss is the same as zero-one loss, and we're right back in the setting where the NFL theorems apply.) Rather than consider $P(c \mid m)$ directly, consider the functional of it, $E(C \mid m)$. For quadratic L(., .), $E(C \mid m) = E((y_H - y_F)^2 \mid m)$. Expanding, we get

$$E(C \mid m) = E((y_H)^2 \mid m) + E((y_F)^2 \mid m) - 2\, E(y_H\, y_F \mid m).$$

$E((y_F)^2 \mid m)$ is some constant, independent of the learning algorithm, determined by the geometry of Y, the likelihood, etc. $E((y_H)^2 \mid m)$ does depend on the learning algorithm, and is determined uniquely by $P(y_H \mid m)$ (in accord with theorem (15)).

The only term left to consider is the correlation term $E(y_H\, y_F \mid m)$. Write this term as $\int dy_F\, y_F\, E(y_H \mid y_F, m)\, P(y_F \mid m)$. Now for the assumptions going into theorem (15), $E(y_H \mid y_F, m) = E(y_H \mid m)$.[5] Accordingly, there is no expected correlation between y_H and y_F: $E(y_H\, y_F \mid m) = E(y_H \mid m) \times E(y_F \mid m)$. This means in turn that our correlation term only depends on the learning algorithm through $E(y_H \mid m)$, which in turn is uniquely determined by $P(y_H \mid m)$. Therefore the theorem is corroborated; $E(C \mid m)$ only depends on the learning algorithm through $P(y_H \mid m)$.[6]

Now $P(c \mid m) = \Sigma_f\, P(c \mid f, m)\, P(f \mid m)$. Therefore $P(c \mid m)$ for a uniform P(f) is proportional to

4. NON-HOMOGENOUS LOSS FUNCTIONS

i) Overview

This section considers the case where L(., .) is not homogenous, so that the NFL theorems do not apply, and therefore we usually can make *a priori* distinctions between algorithms. This section is not meant to be an exhaustive analysis of such loss functions, but rather to illustrate some of the properties and issues surrounding such functions.

An example of such an L(., .) is the quadratic loss function, $L(a, b) = (a - b)^2$ for finite Y. For the quadratic loss function (and in general for any convex loss function when Y is finite), everything else being equal, an algorithm whose guessed Y values lie away from the boundaries of Y is to be preferred over an algorithm which guesses near the boundaries. In addition, consider using a quadratic loss function with two learning algorithms, $P_1(h \mid d)$ and $P_2(h \mid d)$. Construct the algorithms to be related in the following manner: algorithm 2 is a deterministic algorithm that makes the single-valued guess given by the expected y_H guessed by algorithm 1, in response to the training set d and test set question q at hand. (More formally, $P_2(h \mid d) = \delta(h - h^*)$, where $h^*(q, y) = \delta(y, E_1(y_H \mid q, d)) = \delta(y, \Sigma_{y'} [y' \int dh' \, h'(q, y') \, P_1(h' \mid d))])$, where h' is a dummy h argument and y' a dummy y argument.) It is not hard to show [Wolpert 1994a] that for all d and q, $E_2(C \mid q, d) \leq E_1(C \mid d, q)$, i.e., the expected cost for algorithm 2 is always less than or equal to that of algorithm 1, regardless of properties of the target.[4] This holds even for OTS error, so long as the loss function is quadratic.

None of this means that the intuition behind the NFL theorems is faulty for non-homogenous L(., .). However it does mean that that intuition now results in theorems that are not so sweeping as the NFL theorems. In essence, one must "mod out" the possible *a priori* advantages of one algorithm over another that are due simply to the fact that one or the other of those algorithms tends to produce y_H values that are favored *a priori* by the L(., .) at hand.

The primary tool for deriving these less-sweeping results is to compare the OTS behavior of a learning algorithm P(h | d) to that of its "scrambled" or "randomized" version, in which $P(y_H)$ is preserved, but the relationship between the training set d and the hypothesis h is randomly scrambled. Such comparisons show that all the *a priori* advantages conferred on a particular learning algorithm by a non-homogenous L(., .) are due simply to the clumping of $P(y_H)$ in regions that, due to L(., .) are "good". (An example of such a region is the region equidistant from the borders of Y, for a quadratic L(., .).) In particular, according to these comparisons, there is no extra advantage conferred by how the learning algorithm chooses to associate y_H's with particular training sets d and test set inputs q. I.e., there is no advantage conferred by the d-dependence of the algorithm.

Theorem 13 For OTS error, any training set d, and any single-valued target, ϕ, if there is no noise in the generation of the training set, then for the majority and anti-majority strategies A and B,

$$\Sigma_{h_1,h_2} [E(C \mid h_1, h_2, \phi, d, B) - E(C \mid h_1, h_2, \phi, d, A)] \leq 0,$$

where the sum is over all h_1 and h_2 such that both $E(C_{IID} \mid \phi, h_1)$ and $E(C_{IID} \mid \phi, h_2)$ are less than ε.

Theorem 14 For zero-one OTS error, any training set d and any single-valued target, ϕ, if there is no noise in the generation of the training set, then for the majority and anti-majority strategies A and B,

$$\Sigma_{h_1,h_2} [E(C \mid h_1, h_2, \phi, d, B) - E(C \mid h_1, h_2, \phi, d, A)] < 0,$$

where the sum is over all h_1 and h_2 such that both $E(C_{IID} \mid \phi, h_1)$ and $E(C_{IID} \mid \phi, h_2)$ are less than 1.

As usual, since these results hold for any particular training set d, they also hold if one averages over d's generated from ϕ. In addition, similar results hold if A and B refer to cross-validation and anti-cross-validation rather than the majority strategy and the anti-majority strategy. (Rather than sum over all h_1 and h_2 in a certain class, one sums over all generalizers G_1 and G_2 that produce h's that lie in that class.)

The opposite results hold if we instead restrict the hypotheses h_1 to h_2 to lie far from the target ϕ, i.e., if we restrict attention to h's for which $E(C_{IID} \mid \phi, h) > \varepsilon$.[3] It is currently unknown what happens if we adopt other kinds of restrictions on the allowed h's. In particular, it is not clear what happens if one h must be in one region and the other h in a different region.

Say we accept it as empirically true that using the majority algorithm does, in the real world, usually result in lower expected cost than using the anti-majority algorithm, as far as OTS zero-one loss is concerned. Say we accept this as true even in those cases where we use h's that are similar, and therefore have similar goodness-of-fits to f. Given theorems (13) and (14), this implies that there must be some rather subtle relationship between the f's we encounter in the real world on the one hand, and the pairs of h_1 and h_2 that we use the majority algorithm with on the other. In particular, it can't be that the only restriction on those pairs of h_1 and h_2 is that they lie in a sphere of radius ε centered on f. However it might be that certain "reasonable" non-uniform distributions over the set of allowed h's result in our empirical truth. Alternatively, there may be other geometrical restrictions (besides a sphere) that allow h_1 to be close to h_2, but that also allow our empirical truth. As one example, it is conceivable that our empirical truth is allowed if h_1 and h_2 don't fall in a sphere of radius ε centered on f, but rather in a ellipse centered off of f.

ii) Implications of theorems (11) and (12)

It is worth spending a moment to carefully delineate what has been shown here. Theorems (11) and (12) mean that there is no prior P(f) such that, without regard to h_1 and h_2 (G_1 and G_2 in the case of theorem (12)), strategy A is preferable to strategy B, for either of the two sets of strategies considered in those theorems. (Note how much stronger this statement is then saying that averaged over all P(f), two strategies are equal.) At best, strategy A beats strategy B for certain P(f) *and certain associated h_1 and h_2*. Exactly which h_1 and h_2 result in the superiority of strategy A will change with P(f).

So a technique like cross-validation cannot be justified by making assumptions only concerning P(f), nor by making assumptions only concerning G_1 and G_2. Rather one must make assumptions about how G_1 and G_2 correspond to P(f). It is the interplay between all three quantities that determines how well cross-validation performs. (See theorem (1) of section (3) of [Wolpert 1994a].)

Since these results hold for all training sets d, they mean in particular that if f is fixed while d is averaged over, then the two strategies have equal average OTS error. This is important because such a scenario of having f fixed and averaging over d's is exactly where you might presume (due to computational learning theory results) that strategy A beats strategy B.

How can we reconcile the results of this section with those computational learning theory results? Consider the case of theorem (11). My suspicion is what's happening is the following: There are relatively few $\{h_1, h_2\}$ pairs for which strategy A wins, but for those pairs, it wins by a lot. There are many pairs for which strategy B wins, but for those pairs, strategy B only wins by a little. In particular, I suspect that those "many pairs" are the relatively many pairs for which f agrees with h_1 almost exactly as often as it agrees with h_2.

If this is indeed the case, it means that strategy A is unlikely to be grossly in error. Note that this is a confidence-interval statement, exactly the kind that the VC theorems apply to. (However it is a confidence-interval statement concerning *off-training set* error; in this it is actually an extension of VC results, which concern IID error.)

iii) Restricted averages over quantities involving the hypothesis

Since theorem (11) tells us that we must "match" h_1 and h_2 to f for the majority algorithm (A) to beat anti-majority, one might expect that if we sum only over those hypotheses h_1 and h_2 lying close to the target f, then A beats B. Intuitively, if you have a pretty good idea of what f is, and restrict h's to be fairly good approximators of that f, then you might expect majority to beat anti-majority.

Interestingly, the exact opposite is usually true, if one measures how "close" an h is to f as $E(C_{IID} \mid f, h) = \Sigma_{y_H, y_F, q} L(h(q), y_F) f(q, y_F) \pi(q)$. The analysis of this is in appendix B, where in particular the following results are derived.

dation rather than anti-cross-validation. In this sense, cross-validation cannot be justified as a Bayesian procedure. Without restrictions on the set of generalizers under consideration, one cannot say something like "cross-validation is more appropriate for choosing among the generalizers than is anti-cross-validation if one assumes such-and-such a prior".

All of this holds even if we're considering more than two h's at a time (in the case of theorem (11)), or more than two generalizers at a time (in the case of theorem (12)). In addition, theorem (12) holds if we're using any d-determined strategy for choosing between algorithms - nothing in the proof of theorem (12) used any property of cross-validation other than the fact that it uses only d in deciding between the G_i.

A similar result obtains if we simply average over generalizers G without any concern for strategies. With some abuse of notation, we can write such an average as proportional to

$$\int dG\ P(c \mid f, d, G) \propto \int d[P(h \mid d)]\ P(c \mid f, h, [P(h \mid d)]),$$

where "d[P(h | d)]" means the average over the r^n-dimensional simplex that corresponds to the learning algorithms' possible generalizations from training set d.

We can expand our integral as

$$\int d[P(h \mid d)]\ \Sigma_h P(h \mid d)\ P(c \mid f, h, d).$$

By symmetry, this is proportional to $\Sigma_h P(c \mid f, h, d)$. We therefore have the following corollary of lemma (3):

Corollary 4 For OTS error, the uniform average over all generalizers G of P(c | f, d, G) is a fixed function of c, independent of f and d.

(Of course, this in turn results in an identical result for the distribution P(c | f, m, G).)

This has some rather peculiar implications. It means, for example, that for fixed targets f_1 and f_2, if f_1 results in better generalization with the learning algorithms in some set S, then f_2 must result in better generalization with all algorithms not in S. In particular, if for some favorite learning algorithm(s) a certain "well-behaved", "reasonable" f results in better generalization than does a "random" f, then that favorite f results in *worse than random* behavior for all remaining algorithms. For example, let f be a straight line, and S any set of algorithms that generalize well for that target. Then the algorithms not in S have *worse than random* generalization on that f.

Now consider the quantity $\Sigma_{h_1,h_2} P(c \mid f, h_1, h_2, d, i)$, where the variable i is either A or B, depending on which strategy we're using. This quantity tells us whether A or B is preferable, if we uniformly average over all pairs of h's one might use with A and/or B. (Non-uniform averages are considered below.) As such, it is a measure of whether A is preferable to B or vice versa. In appendix A the following result is derived:

Theorem 11 For an OTS homogenous symmetric error, $\Sigma_{h_1,h_2} P(c \mid f, h_1, h_2, d, A) = \Sigma_{h_1,h_2} P(c \mid f, h_1, h_2, d, B)$, for all f and d.

So even if the likelihood is not vertical, averaged over all hypotheses h_1 and h_2, strategy A equals strategy B.

The same reasoning can be used to equate cross-validation with anti-cross-validation. Let G_1 and G_2 be two learning algorithms, and let strategies A and B now refer to cross-validation and anti-cross-validation respectively. Since we are only allowing single-valued h's, and there are r^n such h's, any generalizer is a mapping from a training set d to a point on the r^n-dimensional unit simplex. Therefore any generalizer is a point in the Δ-fold Cartesian product of such simplices, where Δ is the number of possible training sets. (Δ is set by n, r, and the range of allowed m and m'.) So we can talk about averaging over generalizers.

Consider uniformly averaging $P(c \mid f, G_1, G_2, d, i)$ over all G_1 and G_2. Since the training set d is fixed, this corresponds to uniformly averaging over all possible hypotheses h_1 and h_2 constructed from d, and an independent average over all possibilities of whether it is the hypothesis labelled h_1 or the one labelled h_2 that corresponds to the generalizer with lower cross-validation error. (That "independent average" is set by the behavior of the G_i over the proper subsets of d that determine cross-validation error over d.) Consider just the inner-most of these two sums, i.e., without loss of generality say that it is h_1 that corresponds to strategy A and h_2 that corresponds to strategy B. Then by lemma (3), we deduce the following.

Theorem 12 For an OTS homogenous symmetric error, the uniform average over all generalizers G_1, G_2 of $P(c \mid f, G_1, G_2, d,$ cross-validation) equals the uniform average over all G_1 and G_2 of $P(c \mid f, G_1, G_2, d,$ anti-cross-validation).

So for any f, averaged over all generalizers, cross-validation performs the same as anti-cross-validation. In this sense, if one does not restrict the set of generalizers under consideration, then *regardless of the target*, cross-validation is no better than anti-cross-validation. The immediate implication is that without such a restriction, there is no prior P(f) that justifies (in the manner considered in this section) the use of cross-vali-

function L, then there are likely to be differences between the raw averages Σ_f E(C | f, m, A) and Σ_f E(C | f, m, B) for an associated non-homogenous loss function L'.

3 AVERAGING OVER GENERALIZERS RATHER THAN TARGETS

In all of the discussion up to this point, we have averaged over entities concerning targets (namely f, ϕ, or P(ϕ)) while leaving entities concerning the hypothesis (e.g., the generalizer P(h | d)) fixed. Although the results of such an analysis are illuminating, it would be nice to have alternative results in which one doesn't have to specify a distribution over f/ϕ/P(ϕ). Such results would be prior-independent.

One way to do this is to hold one of f/ϕ/P(ϕ) fixed (though arbitrary), and average over entities concerning the hypothesis instead. It is not clear if one can formulate this approach in so sweeping a manner as the NFL theorems, but even its less sweeping formulation results in some interesting conclusions. In that they are prior-independent "negative" conclusions, these conclusions constitute a first principles proof that, for example, cross-validation is non-Bayesian. I.e., they constitute a proof that there is no f/ϕ/P(ϕ) for which cross-validation will necessarily work, independent of the learning algorithms it's choosing among.

i) Averages over entities concerning the hypothesis

Consider the following situation: the target f is fixed, as is the training set d. There are two hypothesis functions, h_1 and h_2. For simplicity, I will restrict attention to the (most popular) case where h's are single-valued, so expressions like "h(x)" are well-defined. Consider two strategies for choosing one of the h_i:

A) Choose whichever of the h_i agrees more often with d;

B) Choose whichever of the h_i agrees less often with d.

Note that if the two h's agree with the training set d equally as often, the strategies are equivalent.

To analyze the relative merits of strategies A and B, start with the following lemma, proven in appendix A.

Lemma 3 For all targets f, for all m' < n, for all training sets d having m' distinct elements, and for all sets of m' values for the values of the hypothesis on the elements of d_X, h($x \in d_X$)

$\Sigma_{h(x \notin d_X)}$ P(c | f, h, d)

is the same function of the cost c, for an OTS homogenous loss that is a symmetric function of its arguments.

too large) set of learning algorithms, one can construct a new one that is head-to-head minimax superior to the algorithms in that set.

iv) More general issues concerning head-to-head minimax behavior

The discussion above leads to the question of whether one can construct a fixed learning algorithm that *is* head-to-head minimax superior to all others. (As a possible alternative, it may be that there are "cycles", in which algorithm α is minimax superior to β, and β is to χ, but in addition χ is superior to α.) It seems unlikely that the answer to this question is yes. After all, due to the NFL theorems, the smallest $\max_f E(C \mid f, m)$ can be for our candidate learning algorithm is $\Sigma_c \, c \, \Lambda(c) / r$. However for all targets f, there is at least one learning algorithm that has zero error on that f (the algorithm that guesses that f regardless of the data set). It is hard to reconcile these facts with our hypothesis, especially given that the Σ_c is usually quite large (e.g., for zero-one loss, it's $(r - 1) / r$).

It may be that if one restricts attention to "reasonable" algorithms (presumably defined to exclude the always-guess-f-regardless-of-d algorithm, among others), there is such a thing as an algorithm that is head-to-head minimax superior to all others. This raises the following interesting issue: All learning algorithms that people use are very similar to one another. To give a simple example, almost all such algorithms try to fit the data, but also restrict the "complexity" of the hypothesis somehow. So the picture that emerges is that people use learning algorithms tightly clustered in a tiny section of the space of all algorithms. It may be that there is an algorithm that is minimax superior to all those in the tiny cluster, even if there is no such thing as a universally head-to-head minimax superior algorithm. If that is the case, then it would be possible to construct an algorithm "superior" to those currently known. But this would only be possible because people have had such limited imagination in constructing learning algorithms.

There are a number of other interesting hypotheses related to minimax behavior. For example, it may be that in many situations not only is cross-validation head-to-head minimax superior to the generalizers it's choosing among, but also superior to anti-cross-validation. As another hypothesis, say we have two algorithms A and B where A is head-to-head minimax superior to B. So $E(C \mid f, m, B)$ "flares out" away from $E(C \mid f, m, A)$ for some f. Now change the loss function to be non-homogenous, in such a way that what used to be extremal values of c become much larger. Given the "flaring out" behavior, this may lead to A's being superior to B even in terms of $\Sigma_f E(C \mid f, m)$ (since at those "flare" f, the difference between $E(C \mid f, m, A)$ and $E(C \mid f, m, B)$ has been increased, and there is little change in the difference for those f for which $E(C \mid f, m, B) < E(C \mid f, m, A)$).

In fact, for non-homogenous loss functions there are distinctions between algorithms in terms of $\Sigma_f E(C \mid f, m)$, as is discussed below. The line of reasoning just given builds upon this fact: it suggests that when there are head-to-head minimax differences between algorithms A and B for some homogenous loss

error, β might correctly choose between them, so the expected cost of β is significantly better than that of α for such situations. In other words, it might commonly be the case that when asked to choose between two generalizers A and B in a situation where they have comparable expected cost, cross-validation usually fails, but in those situations where the generalizers have significantly different costs, cross-validation successfully chooses the better of the two.

Similar behavior may hold even when using cross-validation to choose among more than two algorithms. Under such a hypothesis, cross-validation still has the same average OTS behavior as any other algorithm. And there are actually more situations in which it fails than in which it succeeds (!). However under this hypothesis cross-validation has desirable head-to-head minimax behavior; its behavior will never be much worse than the best possible.[2]

So in particular, assume we are in such a scenario, and that whatever f is, the generalizers amongst which we are choosing all perform well for that f. I.e., we believe our generalizers are well-suited to targets f, although we perhaps cannot formalize this in terms of priors over f, etc. Then we are assured that cross-validation will not perform much worse than the best of those generalizers - all of which perform well for the f at hand - and may perform significantly better. (It should be noted though that even if this hypothesis holds, there are still a number of caveats concerning the use of cross-validation, caveats that, unlike the NFL theorems but just like head-to-head minimax behavior, apply regardless of f. See the averaging-over-hypotheses section below.)

Note that such desirable head-to-head minimax behavior would be prior-independent; if it holds for all targets f, then it certainly holds for all P(f). In addition, such behavior would be consistent with the (conventional) view that cross-validation is usually an accurate estimator of generalization performance. It's important to note though that one can explicitly construct cases where cross-validation doesn't have this desirable head-to-head minimax behavior. (See section 8 in [Wolpert 1994a].)

In addition, it's not at all clear why one should pay attention to distributions conditioned on training set size m rather than on the actual training set d at hand. Yet it is only distributions conditioned on m rather than d that can be said to have head-to-head minimax distinctions (of any sort) with regard to targets f. (To see this, for simplicity consider a deterministic learning algorithm. For such an algorithm, $P(c \mid f, d) = \delta(c, \chi(f, d))$ for some function $\chi(., .)$. The immediate corollary of theorem (1) is that for such an algorithm, for any fixed training set d and cost c, the number of targets f resulting in that cost is a constant, independent of the learning algorithm.)

It should also be noted that in this hypothesis concerning cross-validation's head-to-head minimax properties, cross-validation constitutes a different learning algorithm for each new set of generalizers it chooses among. So even if the hypothesis is true, it would not mean that there is a single learning algorithm that is head-to-head minimax superior to all others. Rather it would mean that given any (presumably not

$P_{C_A,C_B|\Phi,M}(1, 0 \mid \phi, m) = 1$. Despite the fact that $P_{C_A,C_B|\Phi,M}(0, 1 \mid h_1, m) = 1$, $\Sigma_\phi P_{C_A,C_B|\Phi,M}(0, 1 \mid \phi, m)$ can equal $\Sigma_\phi P_{C_A,C_B|\Phi,M}(1, 0 \mid \phi, m)$ by having many ϕ for which $P_{C_A,C_B|\Phi,M}(1, 0 \mid \phi, m)$ is greater than 0, but none for which it equals 1 exactly.

More generally, consider any ϕ other than $\phi = h_1$ or $\phi = h_2$. For such a ϕ, there exists at least one x where $\phi(x) \neq h_1(x)$, and at least one x where $\phi(x) \neq h_2(x)$. Then so long as $\pi(x) > 0$ for all x, for any such ϕ there exists a training set d for which $P(d \mid \phi, m) > 0$ and such that both $h_1(.)$ and $h_2(.)$ have disagreements with ϕ over the set of elements in $X - d_X$. (Just choose $d = \{d_X, \phi(d_X)\}$ and choose d_X to not include all of the elements x for which $\phi(x) \neq h_1(x)$ and to not include all of the elements x for which $\phi(x) \neq h_2(x)$.)

Therefore for any such ϕ, for either the majority or anti-majority algorithm, $E(C_{OTS} \mid \phi, m) = \Sigma_d E(C_{OTS} \mid \phi, d) P(d \mid \phi, m) > 0$. Therefore in particular, for any such ϕ, the anti-majority algorithm has $E(C_{OTS} \mid \phi, m) > 0$. Given that it is certainly true that $E(C_{OTS} \mid \phi, m) > 0$ for all other ϕ (those that equal either h_1 or h_2), this means that for the anti-majority algorithm, for no ϕ is it true that $E(C_{OTS} \mid \phi, m) = 0$. Yet on the other hand, we know that for the majority algorithm, there are ϕ such that $E(C_{OTS} \mid \phi, m) = 0$.

So in this scenario, despite the NFL theorems, there exist ϕ for which we expect C_A to be far less than C_B (e.g., for $\phi = h_1$ the difference in expected costs is 1), but none for which the reverse is true. So for no ϕ will you go far wrong in picking algorithm A, and for some ϕ you *would* go far wrong in picking algorithm B. In such a case, in this particular sense, one can say that algorithm A is superior to algorithm B, even without making any assumptions concerning targets.

iii) The NFL theorems, cross-validation, and head-to-head minimax behavior

In paper one it is pointed out that for some pairs of algorithms the NFL theorems may be met by having comparatively many targets in which algorithm A is just slightly worse than algorithm B, and comparatively few targets in which algorithm A beats algorithm B by a lot. This point is also discussed in example (5) just above. When we have two algorithms of this sort, we say that A is "head to head" minimax superior to B.[1]

It is interesting to speculate about the possible implications of head-to-head minimax superiority for cross-validation. Consider two algorithms α and β. α is identical to some algorithm A, and β works by using cross-validation to choose between A and some other algorithm B. By the NFL theorems, α and β must have the same expected cost, on average. However the following might hold for many choices of A, B, the sampling distribution $\pi(x)$, etc.: For most targets (i.e., most f, or most ϕ, or most $P(\phi)$, depending on which of NFL theorem's averages is being examined) A and B have approximately the same expected OTS error, but β usually chooses the worse of the two, so in such situations the expected cost of β is (slightly) worse than that of α. In those comparatively few situations where A and B have significantly different expected OTS

Theorem 10 For the h* learning algorithm, for all targets ϕ such that $\phi(x) = 0$ for more than m distinct x, $E(C_{OTS} \mid \phi, \text{punt}, m) \leq E(C_{OTS} \mid \phi, \text{no punt}, m)$.

Taken together, these results severely restrict how well an algorithm can be said to perform without knowledge of the prior over targets P(f). In particular, they say that any learning algorithm can just as readily perform *worse than randomly* as better. See paper one for a full discussion of this and other implications of these results.

ii) An example

Some examples of the NFL theorems are presented in paper one. Another one, particularly relevant for the discussion in this second paper, is as follows.

Example 5: Return to the scenario of example 1 from paper one: We have no noise (so targets are single-valued functions ϕ), and the zero-one loss L(., .). Fix two possible (single-valued) hypotheses, h_1 and h_2. Let learning algorithm A take in the training set d, and guess whichever of h_1 and h_2 agrees with d more often (the "majority" algorithm). Let algorithm B guess whichever of h_1 and h_2 agrees *less* often with d (the "anti-majority" algorithm). If h_1 and h_2 agree equally often with d, both algorithms choose randomly between them. Then averaged over all target functions ϕ, $E(C \mid \phi, m)$ is the same for A and B.

Consider the case where $\phi = h_1$, and for simplicity have $h_1(x) \neq h_2(x)$ for all x. Then with some abuse of notation, we can write $P(c_A, c_B \mid \phi, m) = \delta((c_A, c_B), (0, 1))$, or $P_{C_A, C_B \mid \Phi, M}(0, 1 \mid \phi, m) = 1$ for short, where c_i refers to the cost associated with algorithm i and "$\delta(a, b)$" is the Kronecker delta function. In other words, algorithm A, the majority algorithm, will always guess perfectly for this target, whereas algorithm B, the anti-majority algorithm, will never get a guess correct.

Now L(., .) takes on only two values in this example. So by the argument in section 3(iv) of paper one, we know that $\Sigma_\phi P(c_A, c_B \mid \phi, m) = \Sigma_\phi P(c_B, c_A \mid \phi, m)$. (This is a stronger statement than the generic NFL theorems, which only say that $\Sigma_\phi P(C_A \mid \phi, m) = \Sigma_\phi P(C_B \mid \phi, m)$.) So the probability "mass" over all ϕ for having $(c_A, c_B) = (\alpha, \beta)$ is equal to the mass for having $(c_A, c_B) = (\beta, \alpha)$. Stated differently, if $\Sigma_\phi P(c_B, c_A \mid \phi, m)$ is viewed as a function over the $\mathbf{R}^2$ space of values of (c_A, c_B), that function is symmetric under $c_A \leftrightarrow c_B$.

This might suggest that all aspects of algorithms A and B are identical, if one considers the space of all possible f. This is not the case however. In particular, the $c_A \leftrightarrow c_B$ symmetry does not combine with the fact that there is a ϕ for which $P_{C_A, C_B \mid \Phi, M}(0, 1 \mid \phi, m) = 1$ to imply that there must be a ϕ for which

Theorem 6 For OTS error, a vertical $P(d \mid \phi)$, homogenous loss L, uniform $P(\phi)$, and a homogenous test-set noise process, $P(c \mid d)$ equals $\Lambda(c) / r$.

Corollary 2 For OTS error, vertical $P(d \mid \phi)$, homogenous loss L, a homogenous test-set noise process, and uniform $P(\phi)$, $P(c \mid m)$ equals $\Lambda(c) / r$.

Theorem 7 Assume OTS error, a vertical $P(d \mid \phi)$, homogenous loss L, and a homogenous test set noise process. Let α index the priors $P(\phi)$. Then the uniform average over all α of $P(c \mid m, \alpha)$ equals $\Lambda(c) / r$.

Theorem 8 Assume OTS error, a vertical $P(d \mid \phi)$, homogenous loss L, and a homogenous test set noise process. Let α index the priors $P(\phi)$. Then the uniform average over all α of $P(c \mid d, \alpha)$ equals $\Lambda(c) / r$.

Corollary 3 Assume OTS error, a vertical $P(d \mid \phi)$, homogenous loss L, and a homogenous test set noise process. Let α index the priors $P(\phi)$, and let $G(\alpha)$ be a distribution over α. Assume $G(\alpha)$ is invariant under the transformation of the priors α induced by relabelling the targets ϕ. Then the average according to $G(\alpha)$ of $P(c \mid m, \alpha)$ equals $\Lambda(c) / r$.

Now define the empirical error

$$s \equiv \Sigma_{i=1}^{m} \Sigma_{y_H} L(y_H, d_{\mathbf{Y}}(i)) \; h(d_{\mathbf{X}}(i), y_H) \; \pi(d_{\mathbf{X}}(i)) \; / \; \Sigma_{i=1}^{m} \; \pi(d_{\mathbf{X}}(i)) \; .$$

As an example, for zero-one loss and single-valued h, s is the average misclassification rate of h over the training set.

Theorem 9 For homogenous L, OTS error, a vertical likelihood, and uniform P(f), $P(c \mid s, d) = \Lambda(c) / r$.

For the purposes of the next result, restrict things so that both hypotheses h and targets are single-valued (and therefore targets are written as functions ϕ), and there is no noise. $\mathbf{Y}$ is binary, and we have zero-one loss. Let the learning algorithm always guess the all 0's function, h*. The "punt signal" is given if $d_{\mathbf{Y}}$ contains at least one non-zero value. (That signal is supposed to indicate that one is unsure about using the fit h*.) Otherwise a "no-punt" signal is given. Then for the likelihood of (1.1), uniform $\pi(x)$, and $n >> m$,

all f" means $\int df\ A(f) / \int df\ 1$. Note that these integrals are implicitly restricted to those f that constitute X-conditioned distributions over Y, i.e., to the appropriate product space of unit-simplices. (The details of this restricting will not matter, because integrals will almost never need to be evaluated. But formally, integrals over f are over a full r^n-dimensional Euclidean space, with a series of Dirac delta functions and Heaviside functions enforcing the restriction to the Cartesian product of simplices.) Similar meanings for "uniformly averaged" are assumed if we're talking about averaging over other quantities, like ϕ or $P(\phi)$.

i) The NFL theorems

In [Wolpert 1992] it is shown that $P(c \mid d) = \int df\ dh\ P(h \mid d)\ P(f \mid d)\ M_{c,d}(f, h)$, where so long as the loss function is symmetric in its arguments, $M_{c,d}(., .)$ is symmetric in its arguments. (See point (11) of the previous section.) In other words, for the most common kinds of loss functions (zero-one, quadratic, etc.), the probability of a particular cost is determined by an inner product between your learning algorithm and the posterior probability. (f and h being the component labels of the d-indexed infinite-dimensional vectors $P(f \mid d)$ and $P(h \mid d)$, respectively.) Metaphorically speaking, how "aligned" you (the learning algorithm) are with the universe (the posterior) determines how well you will generalize.

The question arises though of how much can be said concerning a particular learning algorithm's generalization behavior without specifying the posterior (which usually means without specifying the prior). In paper one the following "no-free-lunch" (NFL) theorems are derived to partly address this issue. These theorems are valid for any learning algorithm.

Theorem 1 For homogenous loss L, the uniform average over all f of $P(c \mid f, d)$ equals $\Lambda(c) / r$.

Theorem 2 For OTS error, a vertical $P(d \mid f)$, and a homogenous loss L, the uniform average over all targets f of $P(c \mid f, m) = \Lambda(c) / r$.

Theorem 3 For OTS error, a vertical $P(d \mid f)$, uniform $P(f)$, and a homogenous loss L, $P(c \mid d) = \Lambda(c) / r$.

Corollary 1. For OTS error, a vertical $P(d \mid f)$, uniform $P(f)$, and a homogenous loss L, $P(c \mid m) = \Lambda(c) / r$.

Theorem 4 For homogenous loss L and a homogenous test-set noise process, the uniform average over all single-valued target functions ϕ of $P(c \mid \phi, d)$ equals $\Lambda(c) / r$.

Theorem 5 For OTS error, a vertical $P(d \mid \phi)$, homogenous loss L, and a homogenous test-set noise process, the uniform average over all target functions ϕ of $P(c \mid \phi, m)$ equals $\Lambda(c) / r$.

The sets **X** and **Y**, of sizes n and r:	The input and output space, respectively.
The set d, of m **X**-**Y** pairs:	The training set.
The **X**-conditioned distribution over **Y**, f:	The target, used to generate test sets.
The **X**-conditioned distribution over **Y**, h:	The hypothesis, used to guess for test sets.
The real number c:	The cost.
The **X**-value q:	The test set point.
The **Y**-value y_F:	The sample of the target f at point q.
The **Y**-value y_H:	The sample of the hypothesis h at point q.
P(h | d):	The learning algorithm.
P(f | d):	The posterior.
P(d | f):	The likelihood.
P(f):	The prior.

If $c = L(y_F, y_H)$, L(., .) is the "loss function".

L is "homogenous" if $\Sigma_{y_F} \delta(c, L(y_H, y_F))$ is independent of y_F.

If we restrict attention to f's given by a fixed noise process superimposed on an underlying single-valued funtion from **X** to **Y,** ϕ, and if $\Sigma_\phi P(y_F \mid q, \phi)$ is independent of y_F, we have "homogenous" noise.

Table 1: Summary of the terms in the EBF.

2. THE NO-FREE-LUNCH THEOREMS AND 'HEAD TO HEAD' MINIMAX DISTINCTIONS

The goal is to address the issue of how F_1, the set of targets f for which algorithm A outperforms algorithm B, compares to F_2, the set of f for which the reverse is true. To analyze this issue, the simple trick is used of comparing the average over f of f-conditioned probability distributions for algorithm A to the same average for algorithm B. The relationship between those averages is then used to compare F_1 to F_2.

Here and throughout this paper, when discussing non-single-valued f's, "A(f) uniformly averaged over

11) The learning algorithm only sees the training set d, and in particular does not directly see the target. So $P(h \mid f, d) = P(h \mid d)$, which means that $P(h, f \mid d) = P(h \mid d) \times P(f \mid d)$, and therefore $P(f \mid h, d) = P(h, f \mid d) / P(h \mid d) = P(f \mid d)$.

iv) The cost and "generalization error"

12) For the purposes of this paper, the cost c is associated with a particular y_H and y_F, and is given by a loss function $L(y_H, y_F)$. As an example, in regression, often we have "quadratic loss": $L(y_H, y_F) = (y_H - y_F)^2$.

L(., .) is "homogenous" if the sum over y_F of $\delta(c, L(y_H, y_F))$ is some function $\Lambda(c)$, independent of y_H (δ here being the Kronecker delta function). As an example, the "zero-one" loss traditional in computational learning theory ($L(a, b) = 1$ if $a \neq b$, 0 otherwise) is homogenous.

13) In the case of "IID error" (the conventional error measure), $P(q \mid d) = \pi(q)$ (so test set inputs are chosen according to the same distribution that determines training set inputs). In the case of OTS error, $P(q \mid d) = [\, \delta(q \notin d_{\mathbf{X}})\, \pi(q) \,] \;/\; [\, \Sigma_q\, \delta(q \notin d_{\mathbf{X}})\, \pi(q) \,]$, where $\delta(z) \equiv 1$ if z is true, 0 otherwise.

Subscripts $_{OTS}$ or $_{IID}$ on c correspond to using those respective kinds of error.

14) The "generalization error function" used in much of supervised learning is given by $c' \equiv E(C \mid f, h, d)$. (Subscripts $_{OTS}$ or $_{IID}$ on c' correspond to using those respective ways to generate q.) It is the average over all q of the cost c, for a given target f, hypothesis h, and training set d.

In general, probability distributions over c' do not by themselves determine those over c nor vice-versa, i.e., there is not an injection between such distributions. However the results in this paper in general hold for both c and c', although they will only be presented for c. In addition, especially when relating results in this paper to theorems in the literature, sometimes results for c' will implicitly be meant even when the text still refers to c. (The context will make this clear.)

15) When the size of **X**, n, is much greater than the size of the training set, m, probability distributions over c'_{IID} and distributions over c'_{OTS} become identical. (Although as mentioned in the previous section, distributions conditioned on c'_{IID} can be drastically different from those conditioned on c'_{OTS}.) This is established formally in appendix B.

(1.1) $$P(d \mid f) = \Pi_{i=1}^{m} \pi(d_{\mathbf{X}}(i)) \; f(d_{\mathbf{X}}(i), d_{\mathbf{Y}}(i))$$

(where $\pi(x)$ is the "sampling distribution"). In other words, under this likelihood d is created by repeatedly and independently choosing an input value $d_{\mathbf{X}}(i)$ by sampling $\pi(x)$, and then choosing an associated output value by sampling $f(d_{\mathbf{X}}(i), .)$, the same distribution used to generate test set outputs. This likelihood is vertical.

As another example, if there is noise in generating training set **X** values but none for test set **X** values, then we usually do not have a vertical P(d | f). (This is because, formally speaking, f directly governs the generation of test sets, not training sets; see appendix A.)

7) The "posterior" usually means P(f | d), and the "prior" usually means P(f).

8) It will be convenient at times to restrict attention to f's that are constructed by adding noise to a single-valued function from **X** to **Y**, ϕ. For a fixed noise process, such f's are indexed by the underlying ϕ.

The noise process is "homogenous" if the sum over all ϕ of $P(y_F \mid q, \phi)$ is independent of y_F. An example of a homogenous noise process is classification noise that with probability p replaces $\phi(q)$ with some other value in **Y**, where that "other value in **Y**" is chosen uniformly and randomly.

iii) The learning algorithm

9) Hypotheses h are always assumed to be of the form of **X**-conditioned distributions over **Y**, indicated by the real-valued function $h(x \in \mathbf{X}, y \in \mathbf{Y})$ (i.e., $P(y_H \mid h, q) = h(q, y_H)$). Equivalently, where S_r is defined as the r-dimensional unit simplex, hypotheses can be viewed as mappings $h: \mathbf{X} \rightarrow S_r$.

Any restrictions on h are imposed by P(f, h, d, c). Here and throughout, a "single-valued" distribution is one that, for a given x, is a delta function about some y. Such a distribution is a single-valued function from **X** to **Y**. As an example, if one is using a neural net to as one's regression through the training set, usually the (neural net) h is single-valued. On the other hand, when one is performing probabilistic classification (as in softmax), h isn't single-valued.

10) Any (!) learning algorithm (aka "generalizer") is given by P(h | d), although writing down a learning algorithm's P(h | d) explicitly is often quite difficult. A learning algorithm is "deterministic" if the same d always gives the same h. Backprop with a random initial weight is not deterministic. Nearest neighbor is.

Note that since d is ordered, "on-line" learning algorithms are subsumed as a special case.

indicated using lower case letters. Note though that some quantities (e.g., the space **X**) are neither random variables nor instantiations of random variables, and therefore their written case carries no significance.

Only rarely will it be necessary to refer to a random variable rather than an instantiation of it. In accord with standard statistics notation, "E(A | b)" will be used to mean the expectation value of A given B = b, i.e., to mean $\int$ da a P(a | b). (Sums replace integrals if appropriate.)

3) The primary random variables are the hypothesis **X**-**Y** relationship output by the learning algorithm (indicated by H), the target (i.e., "true") **X**-**Y** relationship (F), the training set (D), and the real world cost (C).

These variables are related to one another through other random variables representing the (test set) input space value (Q), and the associated target and hypothesis **Y**-values, Y_F and Y_H respectively (with instantiations y_F and y_H respectively).

This completes the list of random variables.

As an example of the relationship between these random variables and supervised learning, f, a particular instantiation of a target, could refer to a "teacher" neural net together with superimposed noise. This noise-corrupted neural net generates the training set d. The hypothesis h on the other hand could be the neural net made by one's "student" algorithm after training on d. Then q would be an input element of the test set, y_F and y_H associated samples of the outputs of the two neural nets for that element (the sampling of y_F including the effects of the superimposed noise), and c the resultant "cost" (e.g., c could be $(y_F - y_H)^2$).

ii) Training sets and targets

4) m is the number of elements in the (ordered) training set d. $\{d_{\mathbf{X}}(i), d_{\mathbf{Y}}(i)\}$ is the set of m input and output values in d. m' is the number of distinct values in $d_{\mathbf{X}}$.

5) Targets f are always assumed to be of the form of **X**-conditioned distributions over **Y**, indicated by the real-valued function $f(x \in \mathbf{X}, y \in \mathbf{Y})$ (i.e., $P(y_F \mid f, q) = f(q, y_F)$). Equivalently, where S_r is defined as the r-dimensional unit simplex, targets can be viewed as mappings $f: \mathbf{X} \rightarrow S_r$.

Any restrictions on f are imposed by P(f, h, d, c), and in particular by its marginalization, P(f). Note that any output noise process is automatically reflected in $P(y_F \mid f, q)$. Note also that the equality $P(y_F \mid f, q) = f(q, y_F)$ only directly refers to the generation of test set elements; in general, training set elements can be generated from targets in a different manner.

6) The "likelihood" is P(d | f). It says how d was generated from f. It is "vertical" if P(d | f) is independent of the values $f(x, y_F)$ for those $x \notin d_{\mathbf{X}}$. As an example, the conventional IID likelihood is

tion cannot be justified as a Bayesian procedure. I.e., there is no prior for which, *without regard for the learning algorithms in question*, one can conclude that one should choose between those algorithms based on minimal rather than (for example) maximal cross-validation error. In addition, it is noted that for a very natural restriction of the class of learning algorithms, one can distinguish between using minimal rather than maximal cross-validation error - and the result is that one should use maximal error (!).

All of the analysis up to this point assumes the loss function is in the same class as the zero-one loss function (which is assumed in most of computational learning theory). Section 4 discusses other loss functions. In particular, the quadratic loss function modifies the preceding results considerably; for that loss function, there are algorithms that are *a priori* superior to other algorithms under averaging over targets. However it is shown here that no algorithm is superior to its "randomized" version, and in this sense one cannot *a priori* justify any particular learning algorithm, even for a quadratic loss function.

Finally, section 5 presents some open issues and future work.

1. THE EXTENDED BAYESIAN FORMALISM

These papers use the Extended Bayesian Formalism [Wolpert 1994a, Wolpert et al. 1995,Wolpert 1992]. In the current context, the EBF is just conventional probability theory, applied to the case where one has a different random variable for the hypothesis output by the learning algorithm and for the target relationship. It is this crucial distinction that separates the EBF from conventional Bayesian analysis, and that allows the EBF (unlike conventional Bayesian analysis) to subsume all other major mathematical treatments of supervised learning like computational learning theory, sampling theory statistics, etc. (See [Wolpert 1994a].)

This section presents a synopsis of the EBF. Points (2), (8), (14) and (15) below can be skipped ina first reading. A quick reference of this section's synopsis can be found in Table 1.

Readers unsure of any aspects of this synopsis, and in particular unsure of any of the formal basis of the EBF or justifications for any of its assumptions, are directed to the detailed exposition of the EBF in appendix A of the first paper.

i) Overview

1) The input and output spaces are **X** and **Y**, respectively. They contain n and r elements respectively. A generic element of **X** is indicated by 'x', and a generic element of **Y** is indicated by 'y'.

2) Random variables are indicated using capital letters. Associated instantiations of a random variable are

INTRODUCTION

Can one get something for nothing in supervised learning? Can one get useful, caveat-free theoretical results that link the training set and the learning algorithm to generalization error, without making assumptions concerning the target? More generally, are there useful practical techniques that require no such assumptions? As a potential example of such a technique, note that people usually use cross-validation without making any assumptions about the underlying target, as though the technique were universally applicable.

This is the second of two papers that present an initial investigation of this issue. These papers can be viewed as an analysis of the mathematical "skeleton" of supervised learning, before the "flesh" of particular priors over targets are introduced. It should be emphasized that the work in these papers is very preliminary; much remains to be done.

The primary mathematical tool used in these papers is off-training set (OTS) generalization error, i.e., generalization error for test sets that contain no overlap with the training set. (In the conventional measure of generalization error such overlap is allowed.) Section 1 of the first paper explains why that is an appropriate tool to use. Paper one then uses that tool to elaborate (some of) the senses in which all learning algorithms are *a priori* equivalent to one another. This is done by comparing algorithms via averages over targets for "homogenous" loss functions and "vertical" likelihoods.

This second paper extends the analysis to other ways of comparing learning algorithms. In particular, it considers averages over other quantities besides targets, and non-homogenous loss functions. This reveals more senses in which all algorithms are identical. But it also reveals some important senses in which there <u>are</u> "assumption-free" *a priori* distinctions between learning algorithms.

Section 1 of this second paper reviews the mathematical formalism used in these papers.

Section 2 reviews the "no free lunch" (NFL) theorems presented in paper one. It is then pointed out that the equivalence of expected errors between learning algorithms stipulated by those theorems does not mean algorithms have equivalent "head to head" minimax properties. Indeed, it may be that (for example) cross-validation is head-to-head minimax superior to anti-cross-validation (the rule saying choose between generalizers based on which has the *worst* cross-validation error). If so, then in that particular sense cross-validation could be *a priori* justified as an alternative to anti-cross-validation.

Of course, the analysis up to this point in these papers does not rule out the possibility that there are targets for which one particular learning strategy works well compared to another one. To address (the non-trivial aspects of) this issue, section 3 of this second paper discusses the case where one averages over hypotheses rather than targets. The results of such analyses hold for all possible priors over targets, since they hold for all (fixed) targets. This allows them to be used to prove, as a particular example, that cross-valida-

THE EXISTENCE OF *A PRIORI* DISTINCTIONS BETWEEN LEARNING ALGORITHMS

by

David H. Wolpert

The Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe, NM, 87501, USA (dhw@santafe.edu)

Abstract: This is the second of two papers that use off-training set (OTS) error to investigate the assumption-free relationship between learning algorithms. The first paper discusses a particular set of ways to compare learning algorithms, according to which there are no distinctions between learning algorithms. This second paper concentrates on different ways of comparing learning algorithms from those used in the first paper. In particular this second paper discusses the associated *a priori* distinctions that do exist between learning algorithms. In this second paper it is shown, loosely speaking, that for loss functions other than zero-one (e.g., quadratic loss), there are *a priori* distinctions between algorithms. However even for such loss functions, it is shown here that any algorithm is equivalent on average to its "randomized" version, and in this still has no first principles justification in terms of average error. Nonetheless, as this paper discusses, it may be that (for example) cross-validation has better head-to-head minimax properties than anti-cross-validation (choose the learning algorithm with the largest cross-validation error). This may be true even for zero-one loss, a loss function for which the notion of "randomization" would not be relevant. This paper also analyzes averages over hypotheses rather than targets. Such analyses hold for all possible priors over targets. Accordingly they prove, as a particular example, that cross-validation cannot be justified as a Bayesian procedure. In fact, for a very natural restriction of the class of learning algorithms, one should use anti-cross-validation rather than cross-validation (!).

In memory of Tal Grossman.